applied sciences

MDPI

Article

R-PointNet: Robust 3D Object Recognition Network for Real-World Point Clouds Corruption

Zhongyuan Zhang, Lichen Lin and Xiaoli Zhi *

School of Computer Engineering and Science, Shanghai University, Shanghai 201900, China
* Correspondence: xlzhi@shu.edu.cn

Abstract: Point clouds obtained with 3D scanners in realistic scenes inevitably contain corruption, including noise and outliers. Traditional algorithms for cleaning point cloud corruption require the selection of appropriate parameters based on the characteristics of the scene, data, and algorithm, which means that their performance is highly dependent on the experience and adaptation of the algorithm itself to the application. Three-dimensional object recognition networks for real-world recognition tasks can take the raw point cloud as input and output the recognition results directly. Current 3D object recognition networks generally acquire uniform sampling points by farthest point sampling (FPS) to extract features. However, sampled defective points from FPS lower the recognition accuracy by affecting the aggregated global feature. To deal with this issue, we design a compensation module, named offset-adjustment (OA). It can adaptively adjust the coordinates of sampled defective points based on neighbors and improve local feature extraction to enhance network robustness. Furthermore, we employ the OA module to build an end-to-end network based on PointNet++ framework for robust point cloud recognition, named R-PointNet. Experiments show that R-PointNet reaches state-of-the-art performance by 92.5% of recognition accuracy on ModelNet40, and significantly outperforms previous networks by 3–7.7% on the corruption dataset ModelNet40-C for robustness benchmark.

Keywords: R-PointNet; 3D deep learning; point clouds; object recognition; real-world application

Citation: Zhang, Z.; Lin, L.; Zhi, X. R-PointNet: Robust 3D Object Recognition Network for Real-World Point Clouds Corruption. *Appl. Sci.* **2024**, *14*, 3649. https://doi.org/10.3390/app14093649

Academic Editors: João M. F. Rodrigues and Mourad Oussalah

Received: 20 February 2024
Revised: 12 April 2024
Accepted: 23 April 2024
Published: 25 April 2024

1. Introduction

Point cloud refers to a collection of points that represent the spatial distribution and surface characteristics of targets in three-dimensional space. It is commonly used to represent three-dimensional data and contains rich information such as XYZ coordinates, color, intensity value, and time [1]. Point clouds can be obtained using 3D scanners or generated through computer-aided design (CAD) models. With the increasing popularity of 3D data acquisition devices, point clouds have garnered significant attention in various fields including object detection, autonomous driving [2,3], and robotics [4,5]. However, due to the factors such as equipment accuracy and environmental conditions, point clouds obtained from realistic scenes using 3D scanners often contain corruption, including noise and outliers. Figure 1 shows an example of a raw real-world scanned point cloud. In real-world applications, the raw point cloud is usually cleaned up by discarding outliers and denoising the remaining points prior to point cloud recognition, in order to enhance recognition accuracy.

Traditional methods for clearing raw point cloud corruption have been researched in numerous cases [6], such as invalid value removal, statistical approaches [7], radius filtering methods, and so forth. Expectedly, no single traditional method for filtering raw point cloud corruption dominates in the recognition task of point clouds. The choice of algorithm typically relies on the specific circumstances of the recognition application or the equipment used for data acquisition. Furthermore, the selection of the appropriate parameters for cleaning point cloud corruption is contingent upon the scene's characteristics, the data in

question, and the algorithm employed. Consequently, the performance of these methods heavily relies on the experience and adaptability of the algorithm to the application.

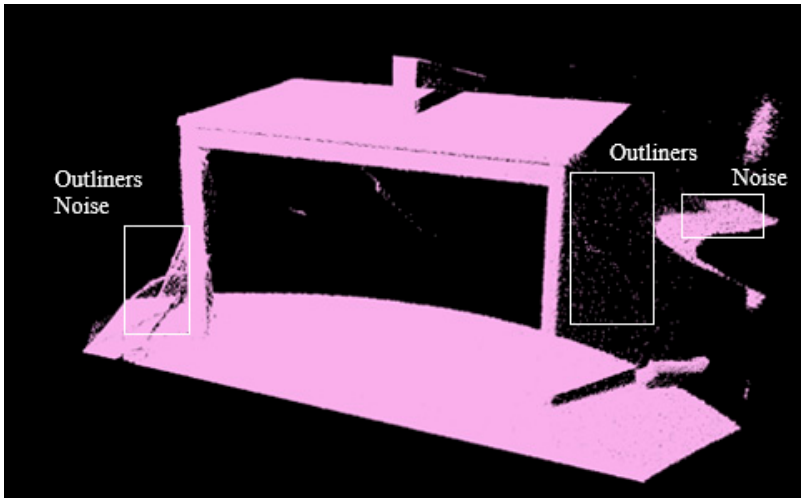

Figure 1. One example of the real-world point clouds with corruption (noise and outliers). Noise is a random error or bias in the point cloud acquisition process and is generally not a real value, outliers are real point cloud data that are not related to the target object.

Since the emergence of deep learning, various deep 3D point cloud recognition architectures have been proposed. Typically, there are three approaches for recognition. The first approach is based on 3D voxel, which convert spatially distributed cluttered point clouds into 3D voxel grids and then recognize them by a well-established 2D or 3D convolutional network. The second approach focuses on multi-view projection images, which project unstructured point clouds into 2D images for classification; and the third utilize point clouds directly. However, both the first and second approaches suffer from the drawback of losing original information through the transformation of point cloud formats. The pioneering work, PointNet [8], addresses this issue by directly operating on raw point clouds and capturing global features for recognition. Furthermore, in order to improve the recognition accuracy and capture local features of the data, PointNet++ [9] was proposed. However, increasing model complexity results in a slight decrease in the robustness of the network when dealing with the corruption of raw point clouds in realistic scenarios.

In this paper, we propose a robust end-to-end point cloud recognition network, named R-PointNet. Our network can capture local discrepancy features through the compensation module, offset-adjustment (OA), and effectively handle noise and outliers in raw point clouds.

Among the various methods used for point cloud recognition, ModelNet40 [10] is considered the most widely accepted benchmark. Over the years, there have been continuous improvements and innovations in network models, leading to enhanced recognition accuracy on ModelNet40. However, these advancements only evaluate model performance from a single perspective using clean data. Given the complexity and significance of 3D point clouds in real-world application, it is crucial to conduct comprehensive robustness benchmarking of point cloud recognition models. In order to better approximate the deficiencies of real-world scanned point clouds and more accurately evaluate our network, we tested R-PointNet on the ModelNet40-C dataset. ModelNet40-C [11], inspired by ImageNet-C, is the first systematic dataset for 3D point cloud recognition robustness benchmark (Figure 2). Many currently representative 3D point cloud recognition models (e.g., PointNet, PointNet++, DGCNN [12], and PCT [13]) exhibit error rates on ModelNet40-C that are nearly three times higher compared to the original ModelNet40.

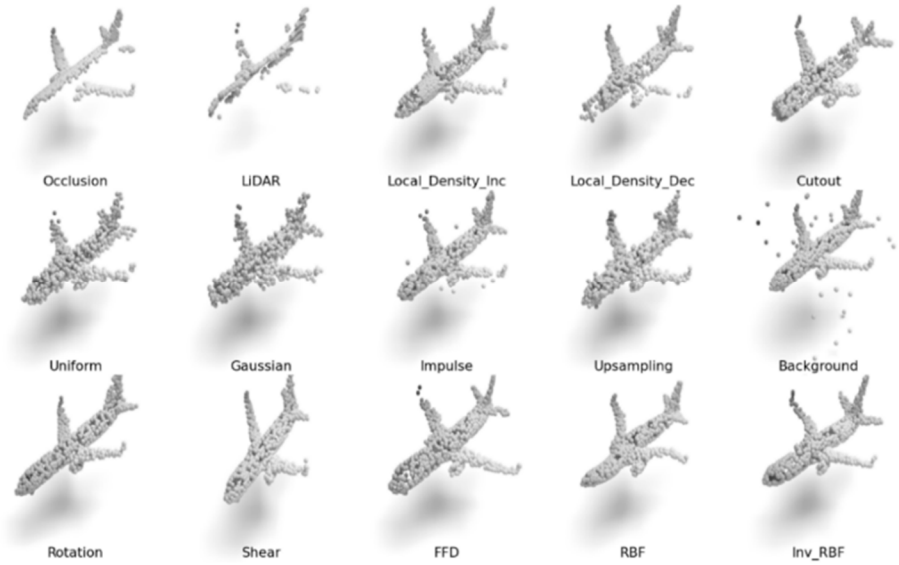

Figure 2. Visualizations of ModelNet40-C. The ModelNet40-C dataset consists of 15 corruption types that represent different out-of-distribution shifts in the real-world applications of point clouds.

Our main contribution can be summarized as follows: (1) We design a compensation module, offset-adjustment (OA), to prevent the influence of corruption of raw point clouds. (2) We propose an end-to-end robust network, R-PointNet, for point cloud recognition. With the proposed OA module, our network is more robust for noise and outliers. (3) We provide an analysis of robustness in similar realistic deficiency scenarios using the proposed R-PointNet and other popular networks.

2. Related Work

2.1. Corruption Processing Method

Raw point clouds usually have corruption, including noise and outliers. Extensive research efforts over the years have resulted in various denoising approaches specifically designed to address different forms of corruption [14,15]. For instance, statistical methods are used to categorize outliers [7], while local smoothing is adjusted using additional input parameters [16], etc. However, these traditional algorithms, while aiming to remove noise, tend to smooth out crucial geometric features, such as sharp boundaries and geometric texture details. As a result, point cloud corruption processing algorithms must strike a balance between smoothing and preserving important features. Furthermore, certain classical algorithms require users to input parameters like search radius, which can significantly impact the results and potentially render parameter-based algorithms less effective if users are uncertain about selecting optimal parameters. These challenges have a detrimental impact on the practical applications of point cloud processing in real-world scenarios.

Deep learning corruption processing methods typically employ a sampling strategy to reduce computational effort. However, most sampling methods are limited by noise sensitivity, are not data-driven [9,17], or ignore spatial distributions [18]. SO-Net [19] utilizes an unsupervised neural network, the self-organizing map (SOM), to exploit the spatial distribution of point clouds. Then, PointNet++ [9] is used for multiple smaller sampling "nodes". However, SO-Net does not incorporate adaptive sampling. Under the assumption of local label consistency, several productions use the geometric center of the voxel grid to represent the sampled points uniformly [20], ignoring differences in the effect of point distribution. These approaches, however, are extremely sensitive to noise and cannot simultaneously understand the spatial distribution of sampled points. TriangleNet [21] is designed to extract features invariant to position, rotation, and scaling disturbances. Although Triangle-Net achieves excellent robustness in the case of extreme corruption, its performance on clean data is not as strong. There are other works to improve

the robustness of models by denoising and upsampling, voting on downsampling point clouds, and exploiting the relative positions of local features.

In summary, traditional methods for cleaning point cloud corruption heavily rely on the expertise and suitability for specific applications, thus posing challenges in handling the noise data of a general nature without requiring additional user intervention. On the contrary, the deep learning approaches offer the advantage of processing corruption without human intervention, but their complexity and lack of accuracy often limit their effectiveness for real-world tasks.

2.2. Point Cloud Recognition Network

With the remarkable advancement of 2D-based computer vision model architectures, there has been a rise in the emergence of 3D point cloud-based recognition methods that involve the proposition of various architectures and operations. Notably, three of the most typical approaches are based on 3D voxels, multi-view projection images, and point clouds [22].

Some early studies converted cluttered point clouds distributed in space into 3D voxel grid representation and then used refined 2D or 3D convolutional networks to achieve shape classification [23]. PointGrid [24], which combined points and voxels, can extract the local features of 3D objects better and achieve great results. However, point clouds suffer from a certain degree of original information loss in the process of voxelization preprocessing due to the resolution. Meanwhile, these methods are highly memory-consuming and generally inefficient in network training with 3DCNN. In contrast, MVCNN [25] takes a set of views of a 3D object by 12 cameras with different viewpoints and then inputs them into the convolutional neural network to obtain the features of each view, and then the features of multi-view images are maximally pooled to obtain the global features and classification. GVCNN [26] first groups different views, then aggregates view features in groups to obtain inter-group features and finally weights different inter-group features by the learned weights to obtain global features before doing classification. Although the multi-view-based approach has achieved good results, there is no definite conclusion about how the number of views should be determined. In addition, multiple views cannot completely cover the 3D objects, especially when the objects have an occlusion situation, which poses a great challenge.

PointNet [8] pioneered extracting each point feature by multilayer perceptron and integrating all point features to obtain the global features of the point cloud with max-pooling, and then performs classification prediction through a fully connected layer, which achieves good classification results. PointNet++ [9] used grouped feature extraction to solve the problem of not being able to capture the local features of point clouds. PointCNN [27] and RSCNN [28] refactor the traditional pyramid CNN to improve the local feature learning for point cloud recognition. DGCNN [12] constructs a dynamic graph of point cloud data based on graph data structure for representation learning. More recently, PCT [13] incorporates Transformer [29] blocks into point cloud learning achieving a state-of-the-art performance.

In conclusion, the current point cloud recognition networks mostly focus on enhancing accuracy on clean datasets by architectural innovations, to some extent ignoring the corruption of real-world point clouds.

3. Method

This paper proposes a robust end-to-end network to deal with unclean point clouds on the basis of PointNet++. In this section, we first detail the advantages of PointNet++ network architecture and ideas for improvement. Thereafter, we show how the offset-adjustment (OA) module enhances network robustness. We then explain the design of R-PointNet.

3.1. PointNet++ Architecture

Innovations in 3D object recognition network architecture have continued to improve the accuracy on point cloud synthesis datasets. Despite this, PointNet++ performs equally well as the latest architectures after controlling factors independent of the network architecture via comprehensive comparison experiments [30]. Moreover, PointNet++ achieved the lowest error rate of classification results on ModelNet40-C among the existing network architectures as described in [11].

As the first object recognition network that directly uses point clouds as input, PointNet attracted widespread attention and improvement. PointNet++ was proposed to improve its poor ability of information integration on the local regions of point clouds. PointNet++ acquires the local features of different regions with sampling and grouping layers. Then, the network integrates the local features of different levels into local–global feature through several step-by-step downsampling operations, named set abstraction.

PointNet++ can be regarded as an encoder–decoder structure, as shown in Figure 3. Encoder is a downsampling process, which achieves multi-level downsampling by several set abstractions to obtain the local features of different regions.

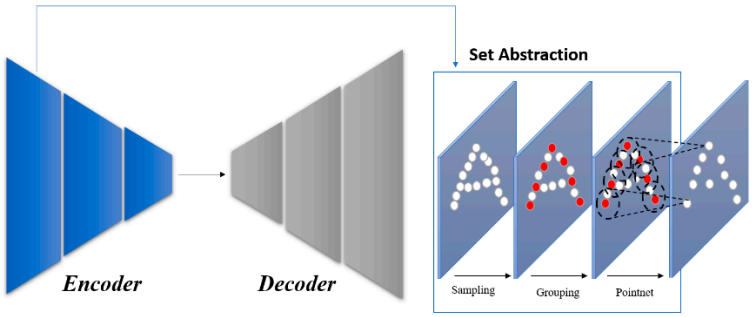

Figure 3. Network structure of PointNet++ (**Left**) and set abstraction (**Right**). PointNet++ obtained multi-level features by set abstraction on the encoder process, and then transmits the local–global feature into several fully connected layers and finally outputs the recognition result on the decoder process.

The set abstraction consists of three layers: Sampling layer, Grouping layer, and PointNet layer. The Sampling layer downsamples the point set with the farthest point sampling (FPS), reducing the input point set to a smaller size. The FPS algorithm makes the sampled points as far away from each other as possible, which has the benefit of making the downsampling results more uniform and allowing the network to cover as many points in space as possible. The sampling process can be understood as selecting N1 key points among N points. The Grouping layer takes each of these key points as the center and finds its neighbors of fixed size (size of K) to form a local neighborhood. The purpose of the grouping layer is to generate N1 local neighborhoods and output their local features. The PointNet layer uses a multi-layer perceptron and max-pooling to extract local features and can be considered a local feature learner. The last output of set abstraction can be considered a global feature. This also means that the network obtains the local–global features of different scales and levels at last.

The decoder of the classification task transmits the local–global feature obtained by the encoder into several fully connected layers, and finally outputs the recognition result with SoftMax.

3.2. Local Operation

The grouping layer of set abstraction is an important part of the PointNet++ architecture. The grouping layer acquires local features by delineating a fixed regional scale, which can have a significant impact on the subsequent point cloud recognition. PointNet++ divides the region by ball query, which is to define a certain radius from the center of the

sampling point and find the points within that radius as neighbors. In order to ensure that the number of sampled points is the same for each local neighborhood, if the number of points in the sampling range is larger than the scale K, then the first K points are directly taken as neighbors; if it is smaller than K, then a point is directly resampled (the closest one to the center) to make up to the scale K. The advantage of ball query is that a fixed area scale is guaranteed. A fixed scale makes local features more generalizable in space and more suitable for the feature extraction of local regions.

However, the search radius is determined by the ball query method: if the realistic point cloud data have serious defects, neighbors near the sampling point will be lost, and the robustness of network will be decreased. To overcome this problem, we can find the K-nearest points in coordinate space to delineate the local region by kNN (K-nearest neighbor point sampling). While a sampled defective point lack neighbors in the local region, kNN will still select the remaining points around the sampling point as neighbors, as shown in Figure 4. Some studies have proven that k-NN generally performs better than ball-query as a local operator in the corruption situation [31]. The reason is that outliers will lose their neighbors in ball-query due to its fixed searching radius, but k-NN will choose neighbors from the remaining points.

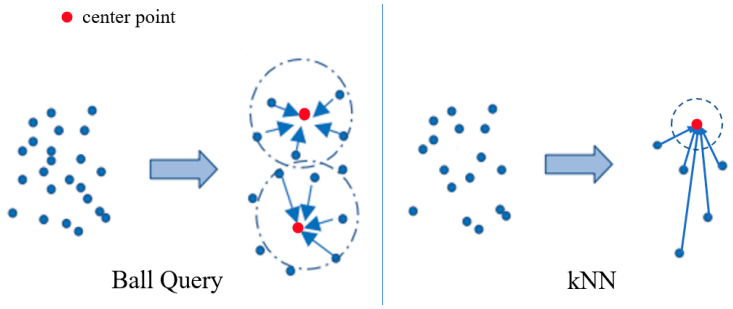

Figure 4. The ball query method takes the sampling point as the center of a certain radius, and selects the points within the radius ball as the neighboring points. kNN (K-nearest neighbor sampling) selects the K nearest points in coordinate space to delineate the local area.

3.3. Offset-Adjustment (OA) Module

Among point cloud recognition tasks, most of the recognition methods based on raw point clouds adopt some sampling algorithms to select points from raw point clouds and then perform local feature learning to recognize point clouds. Most existing sampling algorithms, such as FPS (farthest point sampling), GSS (Gumbel subset sampling), and PDS (Poisson disk sampling), have a common problem: they all may select outliers in raw point clouds, which affect the recognition process. Such a situation is particularly sensitive to real-world data, which also explains the poor performance of the network model in ModelNet40-C.

As mentioned previously, although the FPS algorithm can acquire relatively uniformly sampled points, it generates problems in the realistic applications of point cloud recognition due to the corruption of raw point clouds. This is the reason that we propose to delineate the grouping range by kNN instead of the ball query method in the grouping layer. However, the defective sampling points acquired by the FPS still have a bad impact on the subsequent recognition. To deal with it, we propose the OA module to mitigate the impact of each sampling defect point.

We assume that $T_s \in \mathbb{R}^{P_s*3}$ denotes the set of points obtained from a layer after downsampling by FPS, in which P_s noise points are required to be processed; $E_s \in \mathbb{R}^{P_s*D_l}$ denotes the set of features of the point set after downsampling; p_i and e_i denotes the coordinates and features of a point separately in the point set after the downsampling operation.

First, the OA module obtains T_s and E_s by FPS from the point set of the sampling layer. After that, the nearest points (neighbors) are searched for each sampled point grouped by

kNN, and the k neighbors of sampled point p_i are: $p_{i,1}, \ldots, p_{i,k} \in N(p_i)$; the corresponding set of neighbors' features are $e_{i,1}, \ldots, e_{i,k}$. Then, the group features are updated for all neighbors with the self-attention mechanism [30]. The feature update for neighbor $e_{i,k}$ can be written as

$$e_{i,k} = OA\left(S\left(p_{i,k}, p_{i,j}\right)\xi\left(p_{i,j}\right), \forall p_{i,j} \epsilon N(p_i)\right) \quad (1)$$

where $e_{i,k}$ denotes the feature of the k-th neighbor of the i-th sampled point, and $S\left(p_{i,k}, p_{i,j}\right)$ is a relationship function to calculate the high-level relationship between the neighbors of the sampled point p_i selected after downsampling, which is to find the dot-product similarity of the two points. The formula is as follows

$$S\left(p_{i,k}, p_{i,j}\right) = softmax\left(\phi(e_{i,k})^T \theta\left(e_{i,j}\right) / \sqrt{D'}\right) \quad (2)$$

where ϕ and θ are implemented by one-dimensional convolution, and both are two independent linear transformations; the one-dimensional convolution Conv: $\mathbb{R}^{D_l} \rightarrow \mathbb{R}^{D'}$, D_l and D' are the number of input and output channels, respectively; $\xi\left(p_{i,j}\right) = w_\xi e_{i,j}$ is a linear transformation that serves to perform a dimensional transformation, transforming the dimension from D_l to D'; $OA()$ is an aggregation function, which finally aggregates the features of the k-th neighbor to all other neighbors.

So far, we obtained the updated features $e_{i,j}$ for the k-th neighbor of the i-th sampling point. Then, using the multilayer perceptron and the SoftMax activation function, we can obtain w_p and w_f, which can be expressed as the normalized weights of each coordinate and feature, with the following equations

$$F_p = \left\{\sigma_p(p_{i,k})\right\}_{k=1}^{K},\ W_p = softmax\left(F_p\right), \quad (3)$$

$$F_f = \left\{\sigma_e(e_{i,k})\right\}_{k=1}^{K},\ W_e = softmax\left(F_f\right), \quad (4)$$

where $F_p, F_f,\ W_p,\ W_e \in \mathbb{R}^{K \times 1}$ are the outputs after normalized weights by multilayer perceptron and SoftMax function. Finally, the adjusted coordinates p_i^* and features e_i^* of sampling point p_i are obtained by w_p and w_f, with the following equations

$$p_i^* = W_P^T P,\ P = \left\{p_{i,k}\right\}_{k=1}^{K} \quad (5)$$

$$e_i^* = W_e^T E,\ E = \left\{e_{i,k}\right\}_{k=1}^{K} \quad (6)$$

To put it briefly, the OA module acquires relatively uniform points from the sampling layer, and then uses kNN to find the neighbors of each sampling point and adaptively updates the coordinates and features for each sampling point. Finally, the OA module transmits new coordinates and features to the next layers for processing. Once a defect point is input, the OA module adjusts the input point according to neighbors and shift it to fit the intrinsic geometry shape, which mitigates the effects of defective sampling points.

3.4. R-PointNet

R-PointNet aims to effectively deal with noise and outliers in the raw point clouds. It needs to capture the global feature of point clouds while simultaneously balancing local feature extraction. Therefore, we exploited the advantages of PointNet++, including its encoder–decoder architecture and the operation of set abstraction. The former integrates local features across different regions and levels into the local–global feature, while the latter iteratively extracts features from local regions of the point cloud. The OA module is added to minimize the effect of anomalies in the raw point cloud. Combining the PointNet++ architecture in Section 3.1 and the OA module proposed in Section 3.2, we propose R-PointNet. The network structure is shown in Figure 5.

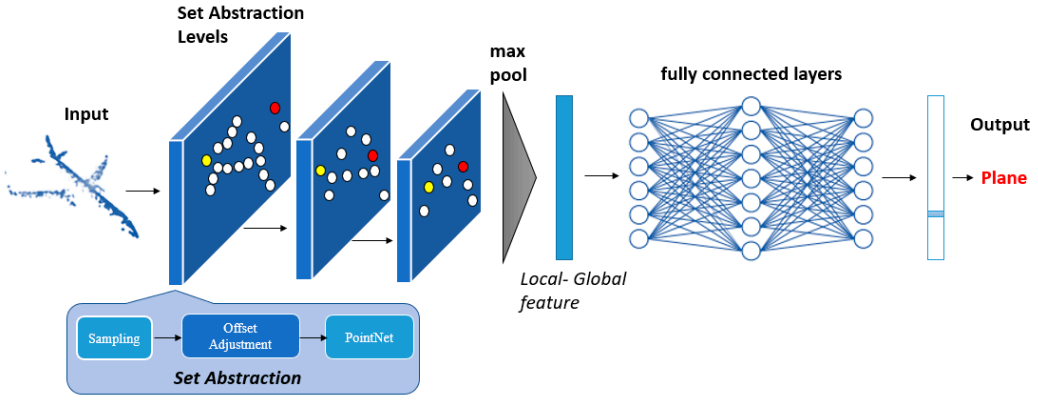

Figure 5. Architecture of our R-PointNet. R-PointNet iteratively extracts the features from local regions of the point cloud and integrates local features across different regions and levels into the local–global feature by set abstraction levels, while the Offset-Adjustment module minimizes the effect of anomalies in the raw point cloud. Then, the muti-level features are integrated into local–global feature. Finally, R-PointNet obtains the recognition results by several fully connected layers.

R-PointNet extracts the features of the point cloud by three layers. The two former layers accomplish downsampling and feature extraction through set abstraction. The third layer concatenates the global features of the two former layers by max-pooling. Then, the extracted feature would be mapped to column vectors by the max-pooling layer to complete the extraction of point cloud features, which means that the network obtains the local–global feature linked by different levels and regions. To achieve the recognition of point clouds, the probability calculation of the features extracted by the network architecture is required. Therefore, a fully connected layer needs to be connected after the max-pooling layer in order to map the learned feature representations to the sample tag space.

4. Experiment and Analysis

Experiments are divided into three parts. First, Section 4.1 provides a detailed experiment process. Second, Section 4.2 analyzes the experimental results and the robustness of the R-PointNet. Finally, Section 4.3 verifies the effectiveness of the OA module by ablation study.

4.1. Experiment Process

The server hardware configuration for all network training is as follows: Ubuntu 20.04 system, 2-core 12-thread Intel Xeon Platinum 8255C processor, 1 NVIDIA RTX 2080 Ti graphics card, Pytorch 1.4.0.

In order to exclude the discrepancy of a training environment among compared networks, we repeated the rest of the networks under the mentioned conditions. During the training period, the augmentation strategy is as follows: random anisotropic scaling in the range $[-0.67, 1.5]$; translation in the range $[-0.2, 0.2]$; and random dropout 20% points. For all network models, the batch size is 16, the training epoch is 200, and the initial learning rate is 0.01.

R-PointNet is tested on two datasets, including ModelNet40, a synthetic dataset commonly used as a point cloud recognition benchmark, and ModelNet40-C, the latest corruption dataset used for robustness benchmark, in order to simulate real-world applications.

ModelNet40 is a point cloud synthesis dataset that is widely used for point cloud recognition benchmark tests. It contains 40 classes with 12,311 point cloud models. For a fair comparison with other networks, we used the official division with 9843 objects for training and 2468 objects for validation. The same sampling strategy as in PointNet++ was used, sampling each object uniformly to 1024 points. ModelNet40-C is the first systematic corruption robustness benchmark for 3D point cloud recognition. It contains 185,000 models

of point clouds with three common categories of distortion: density distortion, noise distortion and transformation distortion. Details are shown in Figure 2.

4.2. Results and Analysis

The test results on ModelNet40 are shown in Table 1. The best recognition accuracy of the R-PointNet network model is 92.5%, which is a 3.2% and 2.1% improvement compared to PointNet and PointNet++, respectively. In Table 1, we can see that our method outperforms almost all state-of-the-art methods on the Modelnet40 benchmark. The only exception is RS-CNN. This is because RS-CNN can learn more meaningful fine shape information such as geometric topological constraints between points. However, RS-CNN does not perform as well on corruption dataset as R-PointNet does.

Table 1. Overall accuracy on ModelNet40 (M40) datasets. "pnt" stands for coordinates of point and "nor" stands for normal vector.

Model	Input	Overall Accuracy (M40)
PointNet [8]	pnt	89.3
PointNet++ [9]	pnt, nor	90.6
RSCNN [28]	pnt	93.6
DGCNN [12]	pnt	92.2
R-PointNet	pnt	**92.5**

We compare our R-PointNet with five representative networks: PointNet, PointNet++, DGCNN, RSCNN, and PCT. These five models stand for different architecture designs of 3D object recognition, and have achieved good accuracy on the clean dataset. In our experiments, the error rate of these popular networks is not 3× larger on ModelNet40-C than ModelNet40 as in the original paper. But compared to the results in Table 1, the error rate still increased by 7–9% under point cloud distortion. The experimental results on ModelNet40-C are shown in Table 2. Combining Tables 1 and 2, the results demonstrate that the point cloud network architecture is still very vulnerable to common distortions. It is worth highlighting that the R-PointNet reaches 89.9% recognition accuracy, which outperforms other networks on Modelnet40-C.

Table 2. Overall accuracy on ModelNet40-C (M40C) datasets. Characterizing the robustness of networks.

Model	Overall Accuracy (M40C)
PointNet	82.2
RSCNN	84.3
DGCNN	84.6
PCT	85.0
PointNet++	86.9
R-PointNet	**89.9**

Finally, we select eight representative corruption types in the ModelNet40-C and test the performance of different networks under different corruption types in detail. The results are in Table 3.

Table 3 presents the detailed accuracy of the six models evaluated on ModelNet40-C separably. As mentioned before, there is a significant decrease in recognition accuracy on each corruption compared to ModelNet40. From the perspective of the model, all networks have different shortcomings in several corruptions, and the performance of our R-PointNet is the most stable across different corruptions, with above-average recognition accuracies, and outperforms all networks in dealing with the noise corruption of the type Uniform and Gaussian.

Table 3. Accuracy of different model architectures on ModelNet40-C (M40C) datasets with different corruption.

Model	Accuracy (M40C)							
	Density Corruption		Nosie Corruption					Transformation Corruption
	Occlusion	LiDAR	Uniform	Gaussian	Impulse	Upsampling	Background	Rotation
PointNet	58.2	**55.6**	90.7	89.6	81.4	89.0	16.5	73.6
RSCNN	58.7	42.1	85.9	86.7	64.2	90.3	**92.5**	81.3
DGCNN	51.3	29.6	88.5	87.0	85.6	91.4	57.4	90.2
PCT	53.9	33.8	93.5	92.1	**85.9**	**93.1**	52.6	**92.1**
PointNet++	55.8	45.0	87.6	86.4	75.4	85.7	82.4	82.9
R-PointNet	**59.2**	50.4	**94.1**	**93.4**	76.1	90.1	86.1	86.1
Average	56.2	42.8	90.1	89.2	78.1	89.9	64.6	84.4

4.3. Ablation Study

We designed an ablation study in order to further illustrate the effectiveness of the proposed OA module. The results of the ablation study are summarized in Table 4.

Table 4. Ablation study on the ModelNet40 (M40) and ModelNet40-C (M40C) validation set. PN, kNN, and OA mean PointNet++ architecture, K-nearest neighbor point sampling and offset-adjustment module.

Model	Ablation	Accuracy (M40)	Accuracy (M40C)
A	PN only	90.6	86.9
B	PN+kNN	90.1	87.2
C	PN+kNN+OA	**91.9**	**90.0**
D	DGCNN	92.2	84.6
E	DGCNN+OA	92.5	88.5

The experiments set model A as baseline. Model A encodes local and global features using PointNet++ architecture. The baseline model A obtains a low accuracy of 90.6%. When we combine model A with kNN (model B), there is a slight decrease in the synthesis dataset, ModelNet40. Compared with kNN, the ball query's local neighborhood guarantees a fixed region scale, thus making the local region feature more generalizable across space, which is preferred for tasks requiring local pattern recognition. But model B is more robust against corruption, considering the improvement in the ModelNet40-C. Finally, when we add the OA module, the model will have a significant improvement on the ModelNet40 and ModelNet40-C datasets (91.9% and 90.0% in model C).

Furthermore, our proposed components with the OA module can directly improve the performance of other architectures on the corruption dataset. When we use DGCNN with our OA module (model E), it will increase the recognition accuracy by 3.9% with its original model (model D) on the ModelNet40-C.

5. Conclusions

Compared to clean datasets, point cloud data in real-world scenarios often contain corruption from noise and outliers. In the field of 3D object recognition, existing networks largely focus on improving the accuracy of clean datasets. Although they have reported promising accuracy on such datasets, their performance is still unsatisfactory in practical application deployment. Hence, object recognition in raw point clouds remains an extremely challenging task.

In this paper, we propose the offset-adjustment (OA) module and build an end-to-end network R-PointNet to deal with real-world datasets. The OA module benefits the feature learning of raw point clouds by adjusting the spatial distribution of sampled defective

points. The adjusted point cloud model aligns more accurately with the intrinsic geometry of the point cloud model. R-PointNet is evaluated on synthetic and real-world datasets through systematic benchmarking and analysis. In the experiments, its accuracy reaches 92.5% on the clean dataset ModelNet40, which is higher than almost any other point cloud recognition network. Additionally, our network outperforms previous networks by 3–7.7% on the corruption dataset ModelNet40-C. The experiments demonstrate that the OA module reduces the effect of erroneous and extraneous points in the sampling and enables the model to learn more comprehensive global features to enhance network robustness.

Our analysis in this work is limited to point cloud recognition, which is an important problem in 3D scene understanding and is an essential component of object detection and retrieval systems. In future work, we intend to apply the methods of this paper to point cloud segmentation to improve the robustness of network in real-world applications.

Author Contributions: Conceptualization, Z.Z.; Methodology, Z.Z.; Validation, Z.Z.; Investigation, L.L.; Writing—original draft, Z.Z.; Writing—review & editing, X.Z.; Supervision, X.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by Shanghai Science and Technology Committee under grant No. 22511106000.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gao, R.; Li, X.; Zhang, J. Recognition of Point Sets Objects in Realistic Scenes. *Mob. Inf. Syst.* **2020**, *2020*, 4925821. [CrossRef]
2. Navarro-Serment, L.E.; Mertz, C.; Hebert, M. Pedestrian Detection and Tracking Using Three-dimensional LADAR Data. *Int. J. Robot. Res.* **2010**, *29*, 1516–1528. [CrossRef]
3. Kidono, K.; Miyasaka, T.; Watanabe, A.; Naito, T.; Miura, J. Pedestrian Recognition Using High-definition LIDAR. In Proceedings of the IEEE Intelligent Vehicles Symposium (IV), Baden-Baden, Germany, 5–9 June 2011; pp. 405–410. [CrossRef]
4. Rusu, R.B.; Blodow, N.; Marton, Z.C.; Beetz, M. Close-range Scene Segmentation and Reconstruction of 3D Point Cloud Maps for Mobile Manipulation in Domestic Environments. In Proceedings of the IEEE RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; pp. 1–6. [CrossRef]
5. Correll, N.; Bekris, K.E.; Berenson, D.; Brock, O.; Causo, A.; Hauser, K.; Okada, K.; Rodriguez, A.; Romano, J.M.; Wurman, P.R. Analysis and Observations From the First Amazon Picking Challenge. *IEEE Trans. Autom. Sci. Eng.* **2018**, *15*, 172–188. [CrossRef]
6. Han, X.-F.; Jin, J.S.; Wang, M.-J.; Jiang, W.; Gao, L.; Xiao, L. A review of algorithms for filtering the 3D point cloud. *Signal Process. Image Commun.* **2017**, *57*, 103–112. [CrossRef]
7. Berzal, F. Outlier analysis. *Comput. Rev.* **2014**, *2014*, 55-10.
8. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 77–85. [CrossRef]
9. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS), Red Hook, NY, USA, 4–9 December 2017. [CrossRef]
10. Wu, Z.R.; Song, S.R.; Khosla, A.; Yu, F.; Zhang, L.G.; Tang, X.O.; Xiao, J.X. 3D ShapeNets: A Deep Representation for Volumetric Shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1912–1920. [CrossRef]
11. Sun, J.; Zhang, Q.; Kailkhura, B.; Yu, Z.; Xiao, C.; Morley Mao, Z.J. Benchmarking Robustness of 3D Point Cloud Recognition Against Common Corruptions. *arXiv* **2022**, arXiv:2201.12296. [CrossRef]
12. Wang, Y.; Sun, Y.B.; Liu, Z.W.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic Graph CNN for Learning on Point Clouds. *Acm Trans. Graph.* **2019**, *38*, 12. [CrossRef]
13. Guo, M.H.; Cai, J.X.; Liu, Z.N.; Mu, T.J.; Martin, R.R.; Hu, S.M. PCT: Point cloud transformer. *Comput. Vis. Media* **2021**, *7*, 187–199. [CrossRef]
14. Ord, K. Outliers in statistical data:, V. Barnett and T. Lewis1994, 3rd edition, (John Wiley & Sons, Chichester), 584 pp., £55.00, ISBN 0-471-93094-6. *Int. J. Forecast.* **1996**, *12*, 175–176. [CrossRef]

15. Rousseeuw, P.J.; Hubert, M. Robust statistics for outlier detection. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2011**, *1*, 73–79. [CrossRef]
16. Huang, H.; Wu, S.H.; Gong, M.L.; Cohen-Or, D.; Ascher, U.; Zhang, H. Edge-Aware Point Set Resampling. *ACM Trans. Graph.* **2013**, *32*, 12. [CrossRef]
17. Hermosilla, P.; Ritschel, T.; Vazquez, P.P.; Vinacua, A.; Ropinski, T.; Assoc Comp, M. Monte Carlo Convolution for Learning on Non-Uniformly Sampled Point Clouds. In Proceedings of the 11th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia (SA), Tokyo, Japan, 4–7 December 2018. [CrossRef]
18. Yang, J.C.; Zhang, Q.; Ni, B.B.; Li, L.G.; Liu, J.X.; Zhou, M.D.; Tian, Q.; Soc, I.C. Modeling Point Clouds with Self-Attention and Gumbel Subset Sampling. In Proceedings of the 32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2019; pp. 3318–3327. [CrossRef]
19. Li, J.X.; Chen, B.M.; Lee, G.H. SO-Net: Self-Organizing Network for Point Cloud Analysis. In Proceedings of the 31st IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 9397–9406. [CrossRef]
20. Thomas, H.; Qi, C.R.; Deschaud, J.E.; Marcotegui, B.; Goulette, F.; Guibas, L.J. KPConv: Flexible and Deformable Convolution for Point Clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October−2 November 2019; pp. 6420–6429. [CrossRef]
21. Xiao, C.X.; Wachs, J. Triangle-Net: Towards Robustness in Point Cloud Learning. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 3–8 January 2021; pp. 826–835. [CrossRef]
22. Guo, Y.L.; Wang, H.Y.; Hu, Q.Y.; Liu, H.; Liu LBennamoun, M. Deep Learning for 3D Point Clouds: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 4338–4364. [CrossRef] [PubMed]
23. Maturana, D.; Scherer, S. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–3 October 2015; pp. 922–928. [CrossRef]
24. Le, T.; Duan, Y. PointGrid: A Deep Network for 3D Shape Understanding. In Proceedings of the 31st IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 9204–9214. [CrossRef]
25. Su, H.; Maji, S.; Kalogerakis, E.; Learned-Miller, E. Multi-view Convolutional Neural Networks for 3D Shape Recognition. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 945–953. [CrossRef]
26. Feng, Y.F.; Zhang, Z.Z.; Zhao, X.B.; Ji, R.R.; Gao, Y. GVCNN: Group-View Convolutional Neural Networks for 3D Shape Recognition. In Proceedings of the 31st IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 264–272. [CrossRef]
27. Li, Y.Y.; Bu, R.; Sun, M.C.; Wu, W.; Di, X.H.; Chen, B.Q. PointCNN: Convolution On X -Transformed Points. In Proceedings of the 32nd Conference on Neural Information Processing Systems (NIPS), Montréal, QC, Canada, 3–8 December 2018; Volume 31. [CrossRef]
28. Liu, Y.C.; Fan, B.; Xiang, S.M.; Pan, C.H. Relation-Shape Convolutional Neural Network for Point Cloud Analysis. In Proceedings of the 32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 8887–8896. [CrossRef]
29. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; Volume 30. [CrossRef]
30. Goyal, A.; Law, H.; Liu, B.W.; Newel, A.; Deng, J. Revisiting Point Cloud Shape Classification with a Simple and Effective Baseline. In Proceedings of the 38th International Conference on Machine Learning (ICML), Virtual, 18–21 July 2021. [CrossRef]
31. Ren, J.W.; Pan, L.; Liu, Z.W. Benchmarking and Analyzing Point Cloud Classification under Corruptions. In Proceedings of the 39th International Conference on Machine Learning (ICML), Baltimore, MD, USA, 17–23 July 2022. [CrossRef]